AMENDMENTS TO THE CLAIMS

The following listing of claims replaces all prior versions, and listings, of claims in the captioned patent application:

Listing of Claims:

1.-20. (Canceled)

21. (Previously Presented) A computer readable medium on which is embedded computer software that programmatically generates a markup language document having at least one markup language element, the software comprising:

a plurality of code sections each automatically adding to the markup language document a corresponding markup language element, wherein each code section utilizes one of the following object-oriented classes each of which models a type of markup language element:

an inline class that models a first type of markup language element in which other markup language elements cannot be nested, and which comprises an opening tag, an argument and no closing tag; and

a container class that models a second type of markup language element in which other markup language elements can be nested, and which comprises an opening tag, a closing tag and an argument disposed between the opening and closing tags.

22. (Previously Presented) The computer readable medium of claim 21, wherein the object-oriented classes further comprise a base class defining a parent-child relationship by which a child object is stored within storage space of the child's parent object,

wherein the inline class is an extension of the base class, wherein each object of the inline class is permitted to be a child object and prohibited from being a parent object; and

wherein the container class is an extension of the base class, wherein each object of the container class is permitted to be either a child object or a parent object,

wherein the type of markup language element modeled by an object depends on the parent-child relationship defined for the class which the object is a member.

23. (Currently Amended) The computer readable medium of claim 21, wherein the markup language is selected from a~~from the~~ group consisting of HTML; XML; XHTML; and SGML.

24. (Previously Presented) The computer readable medium of claim 21, wherein the software further comprises:

at least one class that is an extension of the inline class and which, when processed by a browser, produces a corresponding section of a web page comprising a first type of document item.

25. (Previously Presented) The computer readable medium of claim 24, wherein the first type document item is selected from a group consisting of comment text; formatted text; embedded text; an image; an anchor; a paragraph marker; a line break; and a horizontal rule.

26. (Previously Presented) The computer readable medium of claim 21, wherein the software further comprises:

at least one class that is an extension of the container class and which, when processed by a browser, produces a corresponding section of a web page comprising a second type of document item.

27. (Previously Presented) The computer readable medium of claim 24, wherein the second type document item is selected from a group consisting of a bold text item; a horizontally centered item; a table; a subdocument; and a selection list.

28. (Previously Presented) The computer readable medium of claim 21, wherein the software includes a variable declaration section in which is declared a variable defining a data structure that ultimately contains the markup language document.

29. (Previously Presented) The computer readable medium of claim 21, wherein the members of the container class each utilize temporary files to store its information.

30. (Previously Presented) The computer readable medium of claim 21, wherein the software further comprises:

a class that is an extension of the inline class and which utilizes a buffer data structure.

31. (Previously Presented) The computer readable medium of claim 21, wherein child objects of the inline class and the container class can use their parent's storage when it is available.

32. (Currently Amended) A method for providing executable software program for ~~program that~~ automatically generating a mark-up language document, comprising:

developing a program source code on a computer readable medium comprising a plurality of code sections each automatically adding to the markup language document a corresponding markup language element, wherein the markup language elements comprise:

a first type of markup language element in which other markup language elements cannot be nested and which comprises an opening tag, an argument and no closing tag; and

a second type of markup language element in which other markup language elements can be nested and which comprises an opening tag, a closing tag and an argument disposed between the opening and closing tags;

linking the source code to a library in which is packaged source code of the object-oriented classes each of which models either the first type or the second type of markup language element; and

compiling the linked code to produce the executable software program that, when executed, processes a first set of input conditions to automatically generate the markup language document.

33. (Previously Presented) The method of claim 32, wherein developing a program source code comprises:

providing a set of classes each of which models a type of markup language element, comprising:

an inline class that models the first type of markup language element; and

a container class that models the second type of markup language element;

declaring one or more variables of the types defined by the classes or extensions thereof;

calling any functions defined by the classes and extensions thereof; and

coding other parts of the program source code as necessary.

34. (Previously Presented) The method of claim 33, wherein the classes further comprise:

a base class defining a parent-child relationship by which a child object is stored within the storage space of the child's parent object,

wherein the inline class is an extension of the base class, wherein each object of the inline class is permitted to be a child object and prohibited from being a parent object; and

wherein the container class is an extension of the base class, wherein each object of the container class is permitted to be either a child object or a parent object,

wherein the type of markup language element modeled by an object depends on the parent-child relationship defined for the class of which the object is a member.

35. (Currently Amended) The method of claim 32, wherein the markup language is selected from a ~~from the~~ group consisting of HTML; XML; XHTML; and SGML.

36. (Previously Presented) The method of claim 34, wherein the classes further comprises:

at least one class that is an extension of the inline class and which, when processed by a browser, produces a corresponding section of a web page comprising a first document item.

37. (Previously Presented) The method of claim 36, wherein the first document item is selected from a group consisting of comment text, formatted text, embedded text, an image, an anchor, a paragraph marker, a line break and a horizontal rule.

38. (Previously Presented) The method of claim 36, wherein the classes further comprises:

at least one class that is an extension of the container class and which, when processed by a browser, produces a corresponding section of a web page comprising a second document item.

39. (Previously Presented) The method of claim 38, wherein the second document item is selected from a group consisting of a bold text item, a horizontally centered item, a table, a subdocument and a selection list.